

**RECEIVED
CENTRAL FAX CENTER****JUL 25 2005****IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES**In re Application of:
Broussard

Serial No. 09/870,624

Filed: May 31, 2001

For: SYSTEM AND METHOD FOR
REDUCING MEMORY USE
ASSOCIATED WITH THE
GRAPHICAL REPRESENTATION OF
A LIST CONTROLGroup Art Unit: 2173
Examiner: Bonshock, D.Atty. Dkt. No. AUS920010268US1
(5468-08100)

I hereby certify that this correspondence is being transmitted via
facsimile or deposited with the U.S. Postal Service with
sufficient postage as First Class Mail in an envelope addressed
to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA
22313, on the date indicated below:

07/25/2005
Date

Kevin L. Daffer

APPEAL BRIEF**Box AF**
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313

Sir/Madam:

Further to the Notice of Appeal faxed to and received in the U.S. Patent and Trademark Office on May 23, 2005, Appellant presents this Appeal Brief. The Notice of Appeal was filed following mailing of a final Office Action on February 22, 2005. Appellant hereby appeals to the Board of Patent Appeals and Interferences from a final rejection of claims 1-20 in the final Office Action, and respectfully requests that this appeal be considered by the Board.

I. REAL PARTY IN INTEREST

The subject application is owned by International Business Machines Corporation, a corporation having its principal place of business at New Orchard Road, Armonk, New York, 10504, as evidenced by the assignment recorded at Reel 011888, Frame 0554.

II. RELATED APPEALS AND INTERFERENCES

Notices of Appeal have been filed for the following applications, which share a common specification with the application currently on appeal.

09/870,613: Notice of Appeal filed 2/7/05; Appeal Brief filed on or about April 7, 2005.

09/870,614: Notice of Appeal filed 10/26/04; Appeal Brief filed on or about December 20, 2004.

09/870,615: Notice of Appeal filed 9/14/04; Appeal Brief filed on or about November 9, 2004.

09/870,620: Notice of Appeal filed 12/7/04; Appeal Brief filed on or about February 7, 2005.

09/870,621: Notice of Appeal filed 9/24/04; Appeal Brief filed on or about November 23, 2004.

09/870,622: Notice of Appeal filed 8/24/04; Appeal Brief filed on or about October 25, 2004.

Application serial numbers 09/870,613, 09/870,614, 09/870,615 and 09/870,622 share similar cited art references with the present application; however, dissimilar art is cited in the present application and application serial numbers 09/870,620, and 09/870,621. No other appeals or interferences are known which would directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

Claims 1-20 stand finally rejected. No claims have been allowed, withdrawn, objected to or canceled. Claims 1-20 are being appealed. A copy of claims 1-20 as on appeal are included in the Appendix hereto.

IV. STATUS OF AMENDMENTS

No amendments to the claims were filed subsequent to their final rejection. The Appendix hereto therefore reflects the current state of the claims.

V. SUMMARY OF THE INVENTION

Appellant's claimed invention relates to a display system (10, Fig. 1), computer-readable storage device (e.g., memory 18) and method for generating a display image (e.g., Fig. 9). More specifically, the claimed invention relates to a system and method for displaying an image with a platform-independent

interface during a first time, and displaying a substantially identical image with a platform-dependent interface during a second time, dissimilar from the first. By providing a means for displaying the same image with two different API's at dissimilar times, the presently claimed system and method is able to implement an application program across diverse platforms without changing the "look and feel" of the displayed images. *See, e.g., Abstract, Specification, page 13, lines 10-22, and the Title.*

In some embodiments, the presently claimed display system may include a display device (16, Fig. 1), a graphical user interface (GUI, Fig. 1) and a processor (12, Fig. 1). In general, the processor may be adapted to operate from a windows-based operating system (OS 40, Figs. 1, 2 and 9) for executing a software component (e.g., Button 30, Fig. 9) during runtime of an application (APP 28, Figs. 1 and 2). As described in more detail below, the software component may be displayed upon the display device using an OS-independent application program interface (API) during a first time, and using an OS-dependent API during a second time (API 22 of Figs. 1 and 2 may be implemented with an AWT-type or a Swing-type API). *See, e.g., Specification, page 11, line 27 to page 12, line 6; and page 16, line 25 to page 18, line 8.*

In some cases, the executed software component may generate an image (26, Figs. 1 and 9) upon the display device during a first time, where the displayed image is dependent on code within the operating system. As shown in Fig. 9, for example, an image (26) of button (30) may be displayed using the AWT object model (72, Fig. 9). In other cases, the executed software component may generate a substantially identical image (26, Figs. 1 and 9) upon the display device during a second time, where the substantially identical image is independent of code within the operating system. As shown in Fig. 9, for example, the image (26) of button (30) may also be displayed using the Swing object model (74, Fig. 9). However, the "look and feel" of the image and the substantially identical image may be the same regardless of the platform on which the images are displayed. *See, e.g., Specification, page 12, lines 6-28.*

In some embodiments, the presently claimed method for generating a display image may include running an application program (APP 28, Figs. 1 and 2) upon a computer (10, Fig. 1) and under an operating system (OS 40, Figs. 1 and 2). In some cases, the application program may include a first application program interface (API), which is dependent on the operating system (i.e., API 22 of Figs. 1 and 2 may be an AWT-type interface). As such, the method may run the application program using the first interface to display a first image (26, Figs. 2 and 9) upon a display device (16, Fig. 1) of the computer. As shown in Fig. 9, for example, the application program may be run to display an image (26)

of button (30) using AWT object model (72), so that the "look and feel" of the image is dependent on the native resources of the operating system (*see, e.g.,* dotted lines connecting button 30 to OS 40). *See, e.g.,* Specification, page 13, lines 1-4.

In some cases, the method may further include replacing the first interface with a second interface that is substantially independent of the operating system, yet emulates the behavior of at least a part of the first interface (*i.e.,* API 22 of Figs. 1 and 2 may be replaced with a Swing-type interface). In such cases, the application program may be re-run using the second interface to re-display a substantially identical image (26, Figs. 2 and 9) upon the display device. As shown in Fig. 9, for example, the application program may be re-run to re-display the image (26) of button (30) using Swing object model (74), so that the "look and feel" of the image is no longer dependent on the native resources of the operating system (*see, e.g.,* dotted lines connecting button 30 to the library of Swing components 86). By replacing the OS-dependent interface with an OS-independent interface, the second image can now be displayed on substantially any platform, while retaining the "look and feel" of the first image. *See, e.g.,* Specification, page 13, lines 4-8.

In some embodiments, the presently claimed computer-readable storage device may include an operating system (OS 40, Figs. 1 and 2) and an application program (APP 28, Figs. 1 and 2) adapted for executing code of a software component (*e.g.,* button 30, Fig. 9). The application program may execute the code of the software component during a first time and during a second time, where the second time is dissimilar from the first. During the first time, the executed software component may generate a first image (26, Fig. 9), which is dependent on code executing within the operating system. For example, an image (26) of button (30) may be generated using the native resources of OS (40) by using an AWT interface (72). During the second time, the executed software component may generate a substantially identical image (26, Figs. 2 and 9), which is independent of code executing within the operating system. For example, a substantially identical image (26) of button (30) may be generated using the graphical resources of Swing interface (74), instead of those provided by OS (40) via AWT interface (72). Unlike conventional methods for mixing AWT and Swing components, the OS-independent image (*i.e.,* the second image) may be adapted to overwrite the OS-dependent image (*i.e.,* the first image) upon a display screen during the second time. *See, e.g.,* Specification, page 13, lines 10-22.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

1. Whether claims 1-20 are unpatentable under 35 U.S.C. § 103(a) over U.S. Patent No. 6,727,918 to Nason et al. (hereinafter "Nason") in view of a web publication written by Amy Fowler entitled "*Mixing Heavy and Light Components*" (hereinafter "Fowler").

VII. ARGUMENT

The contentions of the Appellant with respect to the ground of rejection presented for review, and the basis thereof, with citations of the statutes, regulations, authorities, and parts of the record relied upon are presented herein for consideration by the Board.

A. Patentability of Claims 1-9:

1. Nason fails to provide teaching, suggestion or motivation for a software component, which when executed: (i) generates a first image upon a display device during a first time, where the first image is independent of the operating system, (ii) generates a second image upon the display device during a second time, where the second image is dependent of the operating system, and (iii) where the first and second images are substantially identical.

Independent claim 1 states in part:

A display system, comprising... a processor coupled between the display and the graphical user interface and adapted to operate from a windows-based operating system for executing a software component during runtime of an application program, wherein the executed software component generates a first image upon the display independent of code within the operating system during a first time and, during a second time, emulates code that, when executed by the processor, generates a second image upon the display dependent on code within the operating system, and wherein the first and second images are substantially identical.

Statements in the Final Office Action suggest that, "[w]ith regard to claim 1, Nason teaches... implementing two different APIs, one independent of [the] OS, one dependent on [the] OS, to generate images (see column 5, lines 18-22 and lines 45-63), and a reading and rewriting of screen display information, where the primary GUI information is maintained by replacing the primary GUI with a secondary GUI (see column 25, lines 27-40)." (Final Office Action, pages 2-3). As such, the Examiner appears to suggest that the secondary GUI of Nason is somehow equivalent to the presently claimed first image, which is displayed independent of code within the operating system during a first time, and that

the primary GUI of Nason is somehow equivalent to the presently claimed second image, which is displayed dependent of code within the operating system during a second time. The Appellants respectfully disagree, for at least the reasons set forth in more detail below.

The presently claimed case not only describes the use of two different APIs for generating images, but more importantly, describes a means for displaying an image with a platform-independent interface during a first time, and displaying a substantially identical image with a platform-dependent interface during a second time. For example, the presently claimed case teaches that, when an application program is run during the first time, a button (30, Fig. 9) may be displayed using a platform-independent interface (e.g., Swing-type API 74). At some dissimilar point in time, the Swing-type API may be replaced with a platform-dependent interface (e.g., AWT-type API 72), such that when the application program is run during the second time, the same button (30) is re-displayed, this time, using the AWT-type API. In this manner, "either the AWT-type API is executed or the Swing-type API is executed, at two dissimilar times, not in series with one another." See, e.g., Specification, page 11, line 27 to page 13, line 22.

In regards to the above-mentioned Office Action statements, Applicants contend that, while generating images with two different APIs is part of what is claimed, the presently claimed case is not limited to such a feature. Instead, present claim 1 specifically states that the image generated during the first time (e.g., the OS-independent image) and the image generated during the second time (e.g., the OS-dependent image) are substantially identical to one another. In addition, when read in light of the Specification, one skilled in the art would understand that the claimed first and second images are displayed at different times (i.e., the first and second times are dissimilar to one another). As described in more detail below, Nason fails to provide teaching or suggestion for these features, among others.

Contrary to the teachings of the presently claimed case, Nason discloses "computer software [for displaying] one or more user interfaces that can coexist with a native user interface provided by the computer system." (Nason, column 1, lines 17-19, emphasis added). In one embodiment, Nason teaches that an "operating system user interface (the native GUI) may be scaled and/or moved to a specific area of the display permitting a parallel (or secondary) GUI to operate in the open area." (Nason, column 2, lines 50-54). In another embodiment, Nason suggests that a secondary GUI may overlay a portion of the native GUI (see, e.g., Nason, column 6, lines 44-54). Therefore, Nason provides various embodiments

for displaying one or more secondary GUIs outside of, or within, the primary desktop display area, so that the secondary GUIs may be displayed along with (i.e., may coexist with) the native GUI.

Even if Nason were to disclose the use of two different APIs (which he does not), Nason would still fail to teach or suggest that an image may be displayed with a platform-independent interface during a first time, and that a substantially identical image (i.e., the same image) may be displayed with a platform-dependent interface during a second time.

Instead, and as noted above, Nason teaches that one or more secondary GUIs (e.g., calendars, calculators, video conferencing applications, phones, etc.) may be displayed along with the native GUI (e.g., a conventional desktop GUI) by positioning the secondary GUIs in the open areas (referred to as "overscan areas") outside of the primary desktop display area, or alternatively, by overlaying the secondary GUIs on top of the primary desktop display area. See, e.g., Nason, column 3, lines 34-51; column 4, lines 4-29 and lines 51-67; column 5, lines 12-44; and FIGS. 6, 17 and 28 of Nason. Upon reading these passages and others, it becomes painfully clear that the software component of Nason displays substantially different images (i.e., secondary and native GUIs) at the same time, so that the images may coexist with one another. This is non-analogous to the limitations actually recited in present claim 1.

In addition, Nason cannot be modified to include the limitations of present claim 1, since Nason fails to even suggest the desirability for doing so. The mere fact that references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the desirability of the combination [or modification]. *In re Mills*, 916 F.2d 680, 16 USPQ2d 1430 (Fed. Cir. 1990); MPEP 2143.01. For example, Nason fails to suggest a desirability for the following scenarios:

1. A native GUI (e.g., a desktop image) may be displayed in a manner independent from the OS during a first time, and a substantially identical image of the native GUI may be displayed in a manner dependent on the OS during a second time;
2. A secondary GUI (e.g., a pop-up or window displayed adjacent to, or on top of, the desktop) may be displayed in a manner independent from the OS during a first time, and a substantially identical image of the secondary GUI may be displayed in a manner dependent on the OS during a second time; or
3. The native and secondary GUIs may be substantially identical images, one of which is generated during a first time, while the other is generated during a second time.

Nason simply provides no motivation that would enable one skilled in the art to modify the teachings of Nason to include a software component adapted for displaying substantially identical images (one independent, and one dependent on the operating system) at dissimilar times. As a consequence, Nason fails to provide teaching, suggest or motivation for the software component, as recited in present claim 1.

2. Fowler cannot be combined with Nason to provide teaching or suggestion for the presently claimed software component.

Further statements in the Final Office Action suggest that "Nason also teaches the use of Java, but doesn't get into the specifics. Fowler teaches a system of mixing two APIs similar to that of Nason (see page 1, paragraphs 1 and 2), but further teaches specifics of Java's interfaces AWT and Swing, where AWT uses heavyweight components (components that associate with native screen resources, and thus are dependent on the operating system), and where Swing uses lightweight components (components that borrow from screen resources of an ancestor, and thus are independent of the operating system) (see page 2, paragraphs 1 and 3)." (Final Office Action, page 3). The Examiner further suggests that since Fowler teaches that "the lightweight components and heavyweight components look substantially identical (see page 7)," it would have been obvious to one of ordinary skill in the art "to modify the system of using two APIs, of Nason, to include the use of the AWT and Swing APIs, as did Fowler." (Final Office Action, pages 3 and 11).

In the above Office Action statements, the Examiner appears to rely on Fowler for disclosing that: (i) images may be generated with lightweight components (i.e., Swing components independent of the operating system), (ii) images may be generated with heavyweight components (i.e., AWT components dependent on the operating system), and (iii) that the images generated with the lightweight and heavyweight components will look substantially identical. As described in more detail below, the allegation that the "lightweight and heavyweight components look substantially identical" cannot be surmised from the teachings actually provided by Fowler.

The Examiner cites page 7 of Fowler in support of the allegation that "the lightweight and heavyweight components look substantially identical." (Final Office Action, pages 3 and 11). However, there is absolutely no mention of the lightweight and heavyweight components looking substantially identical, as claimed by the Examiner, on page 7 of Fowler. Instead, Fowler discloses the various rules that Swing uses to determine which pop-up to use (e.g., a lightweight, mediumweight or heavyweight pop-up), and illustrates a "Mix Popup Test" in which a heavyweight button obscures a lightweight combo

box pop-up window (Fowler, page 7). However, one skilled in the art would never consider the heavyweight button and the lightweight combo box to look substantially identical to one another.

The Appellants realize that the Examiner's evidence of teaching may not be limited to only those passages cited in the Final Office Action. Therefore, the Appellant provides additional evidence in support of the Appellant's contention that the lightweight and heavyweight components actually disclosed by Fowler do not look substantially identical and cannot be relied upon for disclosing the presently claimed limitation "wherein the first and second images are substantially identical."

On pages 2-3, Fowler discloses that "[t]here are some significant differences between lightweight and heavyweight components...," and that "these differences become painfully apparent when you start mixing Swing components with AWT components." Some of the differences mentioned by Fowler include "a lightweight component can have transparent pixels," whereas "a heavyweight component is always opaque," and "a lightweight component can appear to be non-rectangular because of its ability to set transparent areas," whereas "a heavyweight can *only* be rectangular." In addition to differences in "look," Fowler mentions some differences in "feel" between Swing and AWT components. For example, Fowler notes that "mouse events on a lightweight component fall through to its parent," whereas "mouse events on a heavyweight component do not fall through to its parent." (Fowler, page 3). As another difference in "feel," Fowler notes that "when a lightweight component overlaps a heavyweight component, the heavyweight component is always on top, regardless of the relative z-order of the two components." (Fowler, page 3). Therefore, not only do the teachings of Fowler discredit the Examiner's assertion of the lightweight and heavyweight components looking substantially identical, but they also fail to suggest that the lightweight and heavyweight components could "feel" substantially identical. As a consequence, the lightweight and heavyweight components of Fowler are not substantially identical, as presently claimed.

Because Fowler fails to provide teaching or suggestion for the above-mentioned limitation, the teachings of Fowler cannot be combined with those of Nason to overcome the deficiencies therein. In other words, the combined teachings of Nason and Fowler do not provide teaching or suggestion for the above-mentioned limitations of present claim 1 when all words recited in the claim are rightfully considered. It is noted that "all words in a claim must be considered when judging the patentability of that claim against the prior art." *In re Wilson* 424 F.2d. 1382 (CCPA 1970); MPEP 2143.03.

Furthermore, the teachings of Nason and Fowler cannot be modified to include the above-mentioned limitations of present claim 1, since Fowler and Nason each fail to even suggest a desirability for doing so. The mere fact that references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the desirability of the combination [or modification]. *In re Mills*, 916 F.2d 680, 16 USPQ2d 1430 (Fed. Cir. 1990); MPEP 2143.01.

In fact, and as noted in a Response to the Office action mailed 8/13/04, Fowler appears to teach away from displaying an image with an OS-independent interface (e.g., the Swing API) during a first time, and displaying the same image with an OS-dependent interface (e.g., the AWT API) during a second time. It is noted that a *prima facie* case of obviousness may be rebutted by showing that the art, in any material respect, teaches away from the claimed invention. *In re Geisler*, 116 F.3d 1465, 1471, 43 USPQ2d 1362, 1366 (Fed. Cir. 1997); MPEP 2144.05 (III).

For example, due to the inherent benefits of Swing and the many problems encountered when mixing AWT and Swing, Fowler recommends that Swing components be used exclusively, and that AWT and Swing components be mixed only when the desired Swing component is not available (See, e.g., Fowler, page 2). Since Fowler discourages the use of AWT components, there is no motivation to modify the teachings of Fowler for displaying an image with a Swing component during a first time, and displaying the same image with an AWT component during a second time (or vice versa).

In response to the arguments presented above, the Final Office Action issued the following statements: "[t]he applicants' argue that Fowler teaches away from displaying an image with the AWT API during a first time, and displaying the same image with the Swing API during a second time. In response, the examiner respectfully submits that though Fowler warns of problems that may be encountered when mixing AWT and Swing, Fowler teaches, on page 1, paragraph 2, the mixing of AWT and Swing in the same application program, where the teaching of display of the same image with a second API is taught by Nason, in column 25, lines 27-40" (Final Office Action, page 11, emphasis added). The Examiner now appears to suggest that Nason, not Fowler, provides teaching for displaying the same image with a first and second API.

Contrary to the Examiner's suggestions, Nason does not provide teaching or suggestion for displaying the same image (i.e., a substantially identical image) with a first and second API (see above arguments). Since the primary reference (Nason) fails to provide teaching for the claim limitation, and

the secondary reference (Fowler) teaches away from the claim limitation, the references provide no motivation that would enable one skilled in the art to combine or modify the references in a manner consistent with the claim limitations.

For at least the reasons set forth above, Nason and Fowler each fail to provide teaching, suggestion or motivation for all limitations of present claim 1. Therefore, even if Nason and Fowler were combined and/or modified (without sufficient motivation to do so) the combined teachings of the cited art would still fail to provide teaching or suggestion for the software component, as presently claimed.

3. The Examiner has failed to adequately support and/or establish a *prima facie* ground of obviousness.

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all claim limitations. MPEP § 2143. None of these three criteria have been met by the Examiner in the present case. First of all, no suggestion or motivation to modify the teachings of Nason and Fowler can be found within the cited art to teach or suggest the aforementioned limitations of claim 1, as explained above in Argument 2. The criterion of a reasonable expectation of success cannot be met if no teaching, suggestion or motivation exists, because there is then nothing at which to be successful. Finally, Nason and Fowler each fail to teach, and therefore, cannot be combined to teach, all of the limitations of claim 1, as explained above in Arguments 1 and 2. The third criterion recited above has therefore also not been met, and a *prima facie* case of obviousness has not been established.

Conclusion

As explained in Arguments 1-3 above, at least some limitations of claim 1 and, therefore, at least some limitations of claims 2-9, are not taught or suggested by the cited art. Furthermore, there is no teaching, suggestion or motivation to modify the cited art to teach the limitations of these claims. For at least the reasons set forth above, claims 1-9 are patentably distinct over the cited art. Therefore, the § 103(a) rejection of claims 1-9 is asserted to be erroneous.

B. Patentability of Claims 10-17:

1. Nason fails to teach or suggest a method for generating a display image, where the method includes: (i) running an application program to display a first image using a first OS-dependent interface, (ii) replacing the first interface with a second interface that is independent of the operating system (OS), and (iii) re-running the application program to re-display a second image using the second interface, where the second image is substantially identical to the first image (i.e., has the same "look and feel").

Independent claim 10 recites:

A method for displaying an image, comprising: running an application program upon a computer and under an operating system, wherein the application program includes a first interface dependent on the operating system; displaying a first image upon a display of the computer using the first interface; replacing the interface with a second interface that is substantially independent of the operating system yet emulates the behavior of at least a part of the first interface; re-running the application program; and re-displaying a second image upon the display of the computer using the second interface, wherein the second image has substantially the same look and feel as the first image.

As noted above, the presently claimed case provides a means for displaying an image with a platform-dependent interface during a first time, and displaying a substantially identical image with a platform-independent interface during a second time (or vice versa). To do so, a method is provided in which an application program may be run for displaying an image using a first interface (e.g., an AWT-type interface), which is dependent on the operating system. At some time thereafter, the first interface may be replaced with a second interface (e.g., a Swing interface), which is independent from the operating system. If this occurs, the application program can be re-run for displaying a substantially identical image using the second interface. In a particular aspect of the invention, the displayed images will have substantially the same "look and feel," regardless of whether the API used for displaying the images is platform-dependent (in the case of AWT) or platform independent (in the case of Swing). *See*, e.g., Specification, page 11, line 27 to page 13, line 22.

Statements in the Final Office Action suggest that "[w]ith regard to claim 10, Nason teaches implementing two different APIs, one independent of [the] OS, one dependent on [the] OS, to generate images (see column 5, lines 18-22 and lines 45-63), and a reading and rewriting of screen display information, where the primary GUI information is maintained by replacing the primary GUI with a secondary GUI (see column 25, lines 27-40)." (Final Office Action, page 6). As set forth in more detail

below, the Appellants disagree that any teaching or suggestion for the aforementioned claim limitations can be found within the passages cited within the Final Office Action, or anywhere else within Nason.

In column 25, lines 22-40, Nason suggests that "any number of [secondary] GUIs may be positioned in areas not normally considered the conventional overscan area." (Nason, column 25, lines 22-24). For example, Nason states that "a secondary GUI may be positioned in a small square exactly in the center of the normal display" (Nason, column 25, lines 24-27). In order to do so, Nason suggests that "the techniques of reading and rewriting screen display information can be used to maintain the primary GUI information, or portions of it, in an additional memory and selectively on a timed, computed, interactive, or any other basis, replace a portion of the primary GUI with the secondary GUI such as a pop-up, window or any other display space." (Nason, column 25, lines 27-34). As such, Nason discloses a method in which a portion of a primary GUI (e.g., a conventional desktop image) may be overwritten (or replaced) with a secondary GUI (e.g., a pop-up or other window), so that the secondary GUI may be displayed on top of the primary GUI (e.g., in the center of the normal display).

Though Nason may disclose a method in which a portion of a first interface (e.g., a primary GUI) is replaced with a second interface (e.g., a secondary GUI), Nason does not disclose the remaining limitations recited in claim 10. For example, and as set forth in more detail below, Nason does not disclose the method steps of: (i) running an application program to display a first image using a first OS-dependent interface, (ii) replacing the first interface with a second interface that is independent of the operating system (OS), and (iii) re-running the application program to re-display a second image using the second interface, where the second image is substantially identical to the first image (i.e., has the same "look and feel").

For the sake of argument, let us first assume that the primary GUI of Nason (i.e., the native desktop) is somehow equivalent to the presently claimed "first interface," as suggested by the Examiner. If this is true, Nason cannot provide teaching for the method step of "running an application program to display a first image using a first OS-dependent interface," because Nason does not teach or suggest that the primary GUI can be used for displaying images. Instead, and as pointed out by the Examiner, Nason discloses an alternate display content controller (ADCC), which as software, "may be an application running on the computer operating system, or may include an operating system kernel of varying complexity ranging from dependent on the native operating system for hardware system services to a parallel system independent of the native operating system" (Nason, column 5, lines 18-63). As shown in

FIG. 1, the ADCC of Nason "interacts with the computer utility operating system 5B and hardware drivers 5C to control allocation of display space 1 and [to] create and control one or more [secondary] graphical user interfaces", such as network browser (2) and internet pages (2A and 2B) adjacent to the operating system desktop (3). *See, e.g., Nason, column 5, lines 45-51.*

Therefore, Nason specifically discloses that the alternate display content controller (ADCC) – not the primary graphical user interface (GUI) – is used for displaying images (e.g., creating one or more secondary GUIs).

Now, let's assume that the ADCC of Nason is somehow equivalent to the presently claimed "first interface." If this is true, Nason cannot provide teaching for the method step of "replacing the first interface with a second interface that is independent of the operating system (OS)." For example, although Nason discloses that the ADCC (the alleged "first interface") may be implemented in OS-independent or OS-dependent software, Nason does not disclose that the ADCC may be replaced with another interface, or that OS-independent and OS-dependent versions of the ADCC may both be used for displaying images associated with the same application program.

Finally, Nason cannot provide teaching for the method step of "re-running the application program to re-display a second image using the second interface, where the second image is substantially identical to the first image (i.e., has substantially the same "look and feel" as the first image)," Since Nason fails to suggest that the ADCC could be replaced with a "second interface," Nason cannot be relied upon for teaching that the application program may be re-run for re-displaying a second image using a non-existent "second interface." Furthermore, the Appellants arguments presented above for the patentability of claims 1-9 make clear that although the ADCC of Nason may display various images (e.g., images associated with one or more secondary GUIs along with the primary GUI), Nason does not provide teaching or suggestion for the presently claimed first and second images, which are "substantially identical" (as recited in claim 1) and have "substantially the same look and feel" (as recited in claim 10).

For at least the reasons set forth above, Nason fails to provide teaching or suggestion for all limitations of present claim 10.

2. Fowler cannot be combined with Nason to provide teaching or suggestion for the presently claimed method.

Obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so. *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed.Cir. 1988); *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992); MPEP 2143.01. It is also noted that a *prima facie* case of obviousness may be rebutted by showing that the art, in any material respect, teaches away from the claimed invention. *In re Geisler*, 116 F.3d 1465, 1471, 43 USPQ2d 1362, 1366 (Fed. Cir. 1997); MPEP 2144.05 (III).

As noted above, Nason fails to provide teaching or suggestion for the aforementioned method steps of present claim 10. As set forth below, these methods steps are also not taught or suggested by Fowler. Therefore, Fowler cannot be combined with Nason to overcome the deficiencies therein.

Fowler discloses some of the many differences between, and problems associated with, mixing heavyweight AWT components with lightweight Swing components. Like Nason, however, Fowler simply fails to provide any teaching or suggestion for (i) running an application program to display a first image using a first OS-dependent interface, (ii) replacing the first interface with a second interface that is independent of the operating system (OS), and (iii) re-running the application program to re-display a second image using the second interface, where the second image is substantially identical to the first image (i.e., has the same "look and feel"). Though Fowler suggests that images (e.g., images of buttons, labels and combo-boxes) may be displayed using an OS-dependent (AWT) interface and an OS-independent (Swing) interface, Fowler recommends using Swing components exclusively, and therefore, teaches away from the presently claimed method steps. As such, Fowler provides no teaching, suggestion or motivation for the limitations of present claim 10.

Absent any teaching, suggestion, or motivation to do so, Appellants assert that the teachings of Nason and Fowler do not disclose the method as recited in present claim 10, and furthermore, cannot be combined or modified to do so.

3. The Examiner has failed to adequately support and/or establish a *prima facie* ground of obviousness.

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all claim limitations. MPEP § 2143. None of these three criteria have been met by the Examiner in the present case. First of all, no suggestion or motivation to modify the teachings of Nason and Fowler can be found within the cited art references themselves to teach or suggest the aforementioned limitations of claim 10, as explained above in Argument 2. The criterion of a reasonable expectation of success cannot be met if no teaching, suggestion or motivation exists, because there is then nothing at which to be successful. Finally, Nason and Fowler each fail to teach all limitations of claim 10, as explained above in Arguments 1 and 2. The third criterion recited above has therefore also not been met, and a *prima facie* case of obviousness has not been established.

Conclusion

As explained in Arguments 1-3 above, at least some of the limitations recited in claim 10 and, therefore, at least some limitations of claims 11-17, are not taught or suggested by the cited art. Furthermore, there is no teaching, suggestion or motivation within the cited art to modify the cited art to teach the limitations of claim 10. For at least the reasons set forth above, claims 10-17 are patentably distinct over the cited art. Therefore, the § 103(a) rejection of claims 10-17 is asserted to be erroneous.

C. Patentability of Claims 18-20:

1. Nason fails to teach or suggest a computer-readable storage device comprising an application program adapted for executing code of a software component, which: (i) during a first time, generates a first image dependent on code executing within the operating system, (ii) during a second time, generates a second image independent on code executing within the operating system, (iii) where the second image is adapted to overwrite the first image upon a display screen during the second time, and (iv) where the first and second images are substantially identical.

Independent claim 18 recites:

A computer-readable storage device, comprising: an operating system; and an application program adapted for executing code of a software component, which: during a first time, generates a first image dependent of code executing within the operating system; and during a second time, generates a second image independent on code executing within the operating system, wherein the second image is adapted to overwrite the first image upon a display screen during the second time, and wherein the first and second images are substantially identical.

As noted above, Nason fails to provide teaching, suggestion or even motivation for a software component, which when executed, generates a first image dependent of code within the operating system during a first time, and generates a second image independent of code within the operating system during a second time, where the first and second images are substantially identical.

In addition to the above limitations, Nason fails to provide teaching, suggestion or motivation for the additional limitation recited in claim 18. For example, Nason fails to disclose that, during the second time, the second image (i.e., the OS-independent image) is adapted to overwrite the first image (i.e., the OS-dependent image) upon a display screen. The only mention within Nason for one image overwriting another is when a secondary image (e.g., a pop up or window) overlaps or overwrites a portion of the desktop (such as, when placed in the middle of the display screen). However, Nason does not disclose that the native and secondary GUIs could be substantially identical images. Therefore, even though a secondary GUI may overlap or overwrite a portion of the native GUI (when displayed at the same time), the native and secondary GUIs cannot be considered equivalent to the presently claimed "first and second images," which are substantially identical images that are displayed at different times (i.e., during a first time and during a second time).

Accordingly, Nason fails to provide teaching or suggestion for the computer-readable storage device, as recited in present claim 18.

2. Fowler cannot be combined with Nason to provide teaching or suggestion for the presently claimed computer-readable storage device.

Obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so. *In re Pine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed.Cir. 1988); *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992); MPEP 2143.01. It is also noted that a *prima facie* case of obviousness may be rebutted by

showing that the art, in any material respect, teaches away from the claimed invention. *In re Geisler*, 116 F.3d 1465, 1471, 43 USPQ2d 1362, 1366 (Fed. Cir. 1997); MPEP 2144.05 (IIT).

As noted above, Nason fails to provide teaching or suggestion for the aforementioned limitations of present claim 18. As set forth below, these limitations are also not taught or suggested by Fowler. Therefore, Fowler cannot be combined with Nason to overcome the deficiencies therein.

Like Nason, Fowler fails to provide teaching, suggestion or motivation for the additional limitation recited in claim 18. For example, Fowler fails to disclose that an OS-independent image (e.g., an image generated using the Swing interface) could be used to overwrite an OS-dependent image (e.g., an image generated using the AWT interface) upon a display screen. In fact, Fowler specifically states that this cannot be done. For example, Fowler states that when a lightweight (Swing) component overlaps a heavyweight (AWT) component, the heavyweight component is always on top, regardless of the relative z-order of the two components (see, Fowler, page 3, bullet 4). Fowler also states, "[t]he Swing label will never appear to be on top of the AWT label because the Swing label is rendering its contents in the native window of the frame (its first heavyweight ancestor), while the AWT label is rendering its contents in its own native window" (Fowler, page 4, paragraph 1, emphasis added). As a solution to the problem, Fowler recommends, "do not mix lightweight (Swing) and heavyweight (AWT) components within a container where the lightweight component is expected to overlap the heavyweight one." (Fowler, page 4, paragraph 3). Because Fowler teaches away from the aforementioned limitation, Fowler provides no motivation that would enable one skilled in the art to modify the teachings of Fowler to include such a limitation. As a consequence, Fowler provides absolutely no teaching, suggestion or motivation for the limitations of present claim 18.

Absent any teaching, suggestion, or motivation to do so, Appellants assert that the teachings of Nason and Fowler do not disclose the method as recited in present claim 18, and furthermore, cannot be combined or modified to do so.

3. The Examiner has failed to adequately support and/or establish a *prima facie* ground of obviousness.

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings.

Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all claim limitations. MPEP § 2143. None of these three criteria have been met by the Examiner in the present case. First of all, no suggestion or motivation to modify the teachings of Nason and Fowler can be found within the cited art references themselves to teach or suggest the aforementioned limitations of claim 18, as explained above in Argument 2. The criterion of a reasonable expectation of success cannot be met if no teaching, suggestion or motivation exists, because there is then nothing at which to be successful. Finally, Nason and Fowler each fail to teach all limitations of claim 18, as explained above in Arguments 1 and 2. The third criterion recited above has therefore also not been met, and a *prima facie* case of obviousness has not been established.

Conclusion

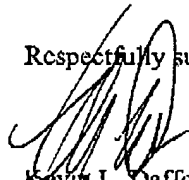
As explained in Arguments 1-3 above, at least some of the limitations recited in claim 18, and therefore, at least some limitations of claims 19-20, are not taught or suggested by the cited art. Furthermore, there is no teaching, suggestion or motivation within the cited art to modify the cited art to teach the limitations of claim 18. For at least the reasons set forth above, claims 18-20 are patentably distinct over the cited art. Therefore, the §103(a) rejection of claims 18-20 is asserted to be erroneous.

IX. CONCLUSION

For the foregoing reasons, it is submitted that the Examiner's rejection of claims 1-20 was erroneous, and reversal of the Examiner's decision is respectfully requested.

The Commissioner is hereby authorized to charge the required fee(s) to deposit account number 50-3268/5468-08100.

Respectfully submitted,



Kevin L. Daffer
Reg. No. 34,146
Attorney for Appellant

Daffer McDaniel, LLP
P.O. Box 684908
Austin, TX 78768-4908
Date: July 25, 2005
JMI

X. APPENDIX

The present claims on appeal are as follows.

1. A display system, comprising:

a display;

a graphical user interface;

a processor coupled between the display and the graphical user interface and adapted to operate from a windows-based operating system for executing a software component during runtime of an application program, wherein the executed software component generates a first image upon the display independent of code within the operating system during a first time and, during a second time, emulates code that, when executed by the processor, generates a second image upon the display dependent on code within the operating system, and wherein the first and second images are substantially identical.

2. The display system as recited in claim 1, wherein said first and second images have the same look and feel.

3. The display system as recited in claim 2, wherein the first and second images comprise pixels presented upon the display via the graphical user interface associated with the application program.

4. The display system as recited in claim 2, wherein the first and second images comprise images of an object selected from a group comprising buttons, list boxes and slide bars on which a pointer device can be directed by a user.

5. The display system as recited in claim 1, wherein the application program is written in Java programming language.

6. The display system as recited in claim 5, wherein said software component comprises a Java application program interface consisting of an abstract windowing toolkit (AWT) during the second time.

7. The display system as recited in claim 5, wherein said software component comprises a Java application program interface consisting of a Swing application program interface during the first time.
8. The display system as recited in claim 1, wherein the operating system comprises a Windows, Unix or OS/2 computer operating system.
9. The display system as recited in claim 1, wherein the first and second images present the same look and feel upon the display independent of the operating system.
10. A method for displaying an image, comprising:

running an application program upon a computer and under an operating system, wherein the application program includes a first interface dependent on the operating system;

displaying a first image upon a display of the computer using the first interface;

replacing the interface with a second interface that is substantially independent of the operating system yet emulates the behavior of at least a part of the first interface;

re-running the application program; and

re-displaying a second image upon the display of the computer using the second interface, wherein the second image has substantially the same look and feel as the first image.
11. The method as recited in claim 10, wherein said displaying and re-displaying comprises presenting pixels upon the display via a graphical user interface associated with the application program.
12. The method as recited in claim 10, wherein said displaying and re-displaying comprises presenting objects selected from a group comprising buttons, list boxes and slide bars upon the display on which a pointer device can be directed by a user.
13. The method as recited in claim 10, wherein the application program is written in Java programming language.

14. The method as recited in claim 10, wherein said running comprises implementing the first interface as a Java application program interface consisting of an abstract windowing toolkit (AWT).

15. The method as recited in claim 10, wherein said re-running comprises implementing the second interface as a Java application program interface consisting of a Swing application program interface that draws the second image over any other image that exists within the area of the display adapted to receive the second image.

16. The method as recited in claim 10, wherein said running and re-running comprises operating the computer on a Windows, Unix or OS/2 operating system.

17. The method as recited in claim 10, wherein the look and feel of the second object is independent of the operating system.

18. A computer-readable storage device, comprising:

an operating system; and

an application program adapted for executing code of a software component, which:

during a first time, generates a first image dependent of code executing within the operating system; and

during a second time, generates a second image independent on code executing within the operating system, wherein the second image is adapted to overwrite the first image upon a display screen during the second time, and wherein the first and second images are substantially identical.

19. The computer-readable storage device as recited in claim 18, further comprising an application interface code that executes, during the first time, separate and apart from the code within the operating system, and wherein said application interface code is written in a Java programming language as a Swing application program.

20. The computer-readable storage device as recited in claim 18, further comprising an application interface code that executes, during the second time, from code within the operating system, and wherein said application interface code is written in a Java programming language as an abstract windowing toolkit (AWT).

IX. EVIDENCE APPENDIX

No evidence submitted pursuant to §§ 1.130, 1.131, or 1.132 of this title has been entered during the prosecution of the captioned case. In addition, no evidence has been entered by the examiner.

X. RELATED PROCEEDINGS APPENDIX

No decisions have been rendered by the Board for the related appeals listed in the Related Appeals and Interferences section of this Appeal Brief.

No other prior and pending appeals, interferences, or judicial proceedings are known to Appellant or Assignee which would directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.